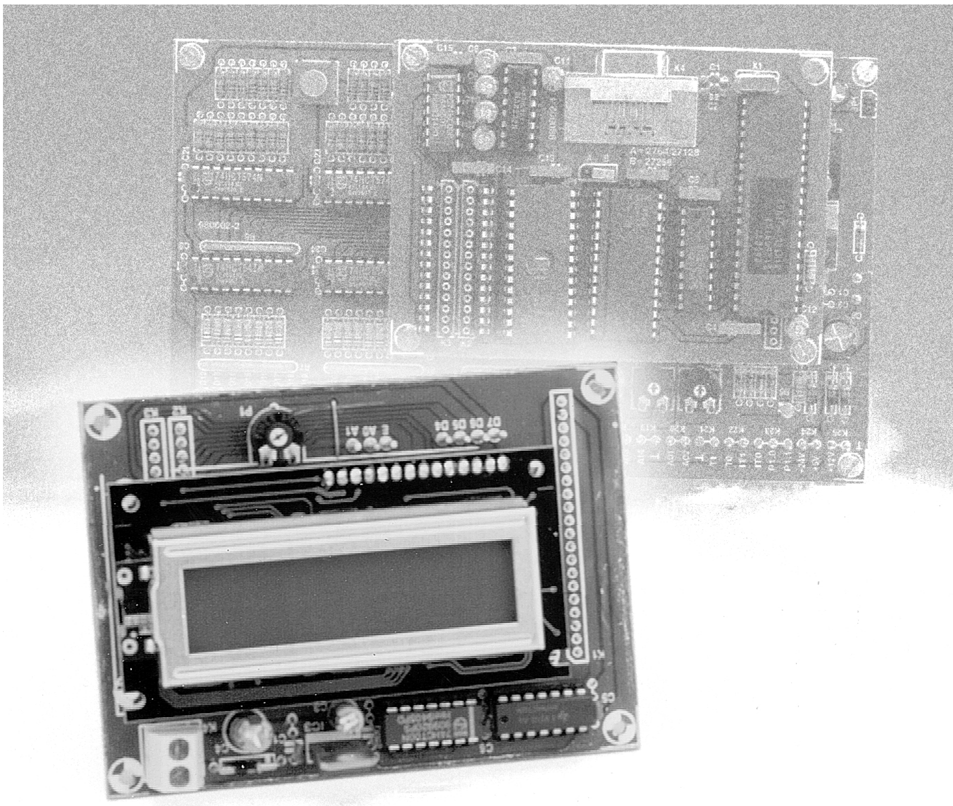
LCD drive

via 8032 BASIC



The 80C32 BASIC control computer published in the February/March 1998 issues of this magazine lends itself to a number of interesting experiments. One of these, described in this article, is its use as direct driver for a liquid crystal display via a 4-bit or 8-bit interface.

Design by A. Köhler

INTRODUCTION

Programmed with the MCS-51 BASIC interpreter, the 80C32 BASIC control computer published in this magazine early last year enables interesting experiments even for those who do not have special knowledge of microcontroller programming.

The single-board computer (SBC) consists of a small main computer board on which the microcontroller is configured, and a larger multifunction extension board to which the main board is fitted. What was not foreseen is a simple, direct connection for a liquid-crystal display (LCD). The present article describes a setup that allows a standard 2×16 character LCD to be connected to the control computer.

The display is driven via the 4-bit interface of the extension board or a discrete 8-bit interface. The latter is no doubt the better solution since only the microcontroller configuration is needed to drive the LCD. Nevertheless, the article gives short BASIC listings for both variants.

4-BIT INTERFACE

Most alphanumeric liquid-crystal displays are based on a member of the HD64780 family of controllers. In most literature on these controllers, the possibility of driving the circuits via four data lines is often not included. Apart from these lines, the control of an LCD also requires three status lines. If a back signal from the display is not needed, it is possible to make do with six control lines.

This small number of lines makes it possible for the LCD unit to be driven via an 8-bit output port. For this purpose, port connections D_{01}–D_{08} (K_{13} to K_{16} on the extension board) are used; note that D_{07} is not used.

Output power driver IC_8 would not be required were it not for the fact that it must support the software for inverting the signals. This is true of the listings for both versions. Listing 1 is for use with the original circuit, whereas with Listing 2 the port lines of IC_8 are replaced by wire bridges. In this case, the LEDs in the port lines are, of course, superfluous.

The programs are annotated liberally, so that a detailed discussion here is not needed.

The control bits (type of display, matrix format, reset, cursor to the left or right, and so on) are set up to line 170 and sent to the display controller via routines 700–780. The bits on the output interface are allocated as shown in the table below.

1

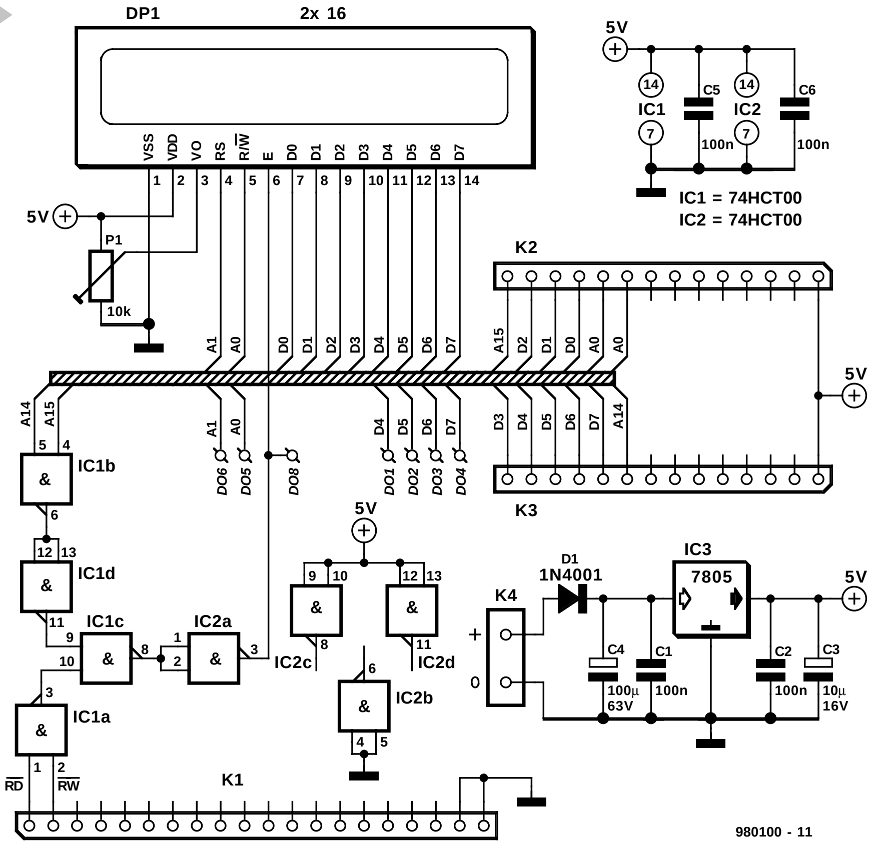


Figure 1. Circuit diagram of the LCD interface. In the 4-bit version, the components associated with regulator IC_3 may be omitted.

Bit	Significance	LCD pin
7 (D_{08})	LCD latch enable(active high)	6
6 (D_{07})	not used	
5 (D_{06})	RS (L = command; H = data)	4
4 (D_{05})	R/$\overline{W}$ (L = write)	5
3 (D_{04})	LCD data bit 7	14
2 (D_{03})	LCD data bit 6	13
1 (D_{02})	LCD data bit 5	12
0 (D_{01})	LCD data bit 4	11

Even though the display is limited to 2 × 16 characters, the controller drives 2 × 40 characters at all times. Since a return line from the display to the SBC is not provided, the software must ensure that after the 2 × 16 visible characters (program lines 250–300) have been input, the invisible character positions in the memory are filled with space characters. This is effected in program lines 310–350.

Finally, the data bits are sent to the controller in routines 400–600: the four most significant first, the least significant last.

It should be ensured that the busy flag is enabled only after the third output of 03H (line 60).

Note the row of 'wait' loops that make any kind of handshaking unnecessary. These loops are generously proportioned and may be shortened if required.

Power for the interface is derived from the SBC.

8-BIT INTERFACE

It is, of course, much neater to link the liquid-crystal display directly to the SBC, for which purpose the computer and display interface have the same types of connectors. The controller then receives, apart from the earlier mentioned control signals RS (address A_1) and R/$\overline{W}$ (address A_0), all eight data bits in parallel. Taking a basic address of 0C000H, the address allocation is

0C000H	write instruction
0C001H	read instruction
0C002H	write data
0C003H	read data

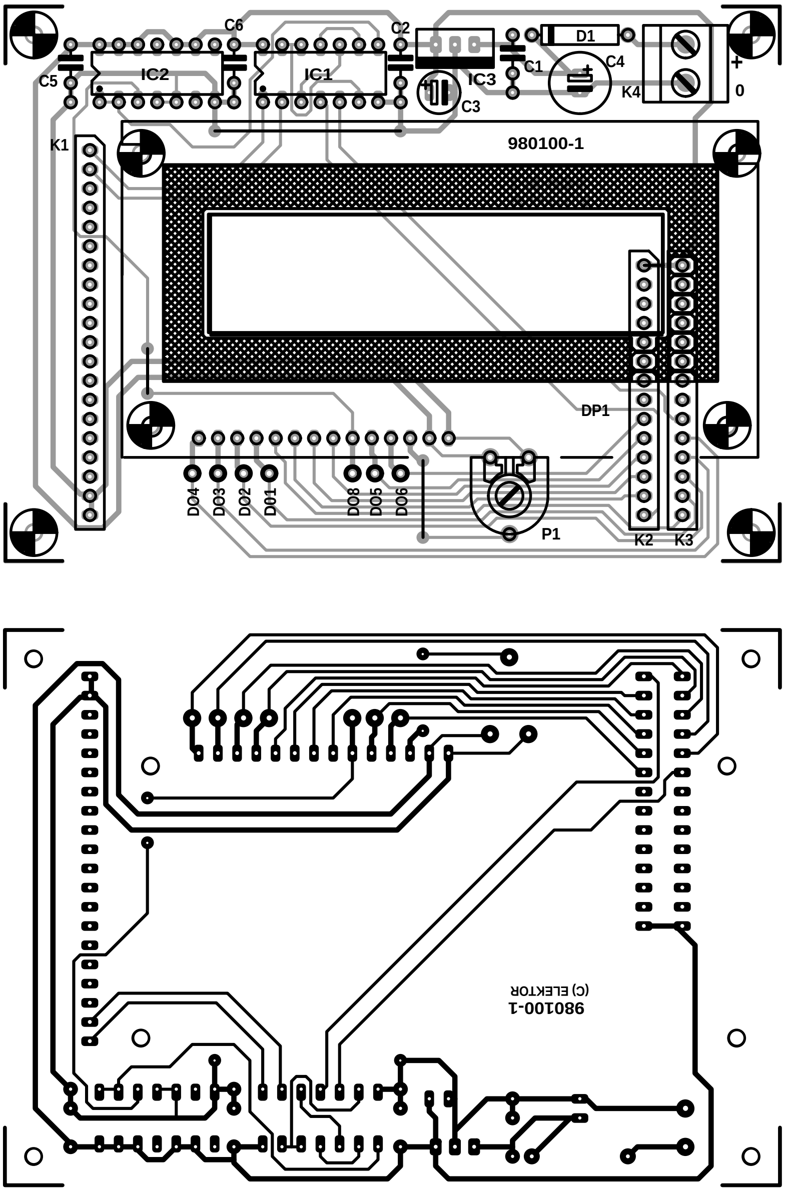
Address 0C00H (enable line at pin 6) is decoded by an AND operation of the $\overline{WR}$ and $\overline{RD}$ signals in the SBC and address lines A_{14} and A_{15}. This arrangement makes the LCD interface multi-addressable, which is particularly useful when other peripheral building blocks are linked to the SBC.

Listing 3 shows that the 8-bit version has clear advantages over the 4-bit version. For instance, there are no wait loops to obstruct the execution of the short, plain program.

Power cannot be taken from the SBC: on the contrary, the controller should now be powered via the LCD interface board. Power is derived from a discrete mains adaptor, whose output is stabilized by IC_3 and smoothed by capacitors C_1–C_4. Diode D_1 protects the circuit against inadvertent incorrect polarity of the supply voltage.

[980100]

2



Parts list
Resistors:
P_1 = 10 kΩ preset potentiometer

Capacitors:
C_1*, C_2, C_5*, C_6* = 0.1 µF
C_3 = 10 µF, 16 V, upright
C_4* = 100 µF, 63 V. upright

Semiconductors:
D_1 = 4001

Integrated circuits:
IC_1*, IC_2* =74HCT00
IC_3* = 7805

Miscellaneous:
K_1* = 20-way SIL socket
K_2*, K_3* = 14-way SIL socket
K_4 = 2-way terminal strip for board mounting
K_5 = 2¥16 character liquid-crystal display (for instance, Sharp Type LM16A211)

* Not used in 4-bit version

Figure 2. The single-sided printed-circuit board for the LCD interface which is not available ready made. The 8-version is linked directly to the SBC board, for which purpose the same connectors are used as on that controller.

```
Listing 1

5   REM ULN 2803 replaced by wire links
10  STRING 100,20   : REM Memory for string
20  K = 3 : GOSUB 700 : REM Init I
30  FOR Z = 1 TO 4000
40  NEXT Z
50  K = 3 :  GOSUB 700 : REM Init 2
60  K = 3 :  GOSUB 700 : REM Init 3
70  K = 2 :  GOSUB 700 : REM Init.4
80  K = 2 :  GOSUB 700 : REM 4-bit interface
90  K = 8 :  GOSUB 700 : REM 2 lines,
    5*7 matrix
100 K = 0 : GOSUB 700 : REM Clear display
110 K = 1 : GOSUB 700
120 K = 0 : GOSUB 700 : REM Cursor home
130 K = 2 : GOSUB 700
140 K = 0 : GOSUB 700 : REM Increment counter,
    do not shift
150 K = 6 : GOSUB 700
160 K = O : GOSUB 700 : REM Display on,
    cursor on, flash
170 K = 15 : GOSUB 700
180 REM ———————————————————-
250 PRINT “ Max. 36 characters may be entered
    up to arrow  <”
260 INPUT “ Please enter text string to be
    displayed :”,$(1)
270 FOR Z = 1 TO 16
280 D=ASC($(1),Z) : REM Individual char’s,
    supplied by user
290 GOSUB 400
300 NEXT Z
310 FOR Y = 16 TO 39 : REM Fill LCD controller
    memory
320 D = 32 : REM not having a display function
330 GOSUB 400
340 NEXT Y :REM with dummy characters
350 GOTO 250
360 REM ———————————————————-
```

```
400 REM Subroutine for data output
410 L = D.AND.15 : REM Data of low part
420 H = D.AND.240 : REM Data of high part
430 H = H/16 : REM Shift high data
440 H = H.OR.32 : REM Enable data
450 XBY(0C000H) = H
460 H = H.OR.160 : REM Enable latch
470 XBY(0C000H) = H
480 H = H.AND.127 :REM Disable latch
490 XBY(0C000H) = H
500 FOR U = 1 TO 100
510 NEXT U
520 L = L.OR.32 : REM Enable data
530 XBY(0C000H) = L
540 L = L.OR.160 : REM Enable latch
550 XBY(0C000H) = L
560 L = L.AND.127 : Disable latch
570 XBY(0C000H) = L
580 FOR U = 1 TO 100
590 NEXT U
600 RETURN
610 REM ——————————————————-
700 REM Subroutine for command outputting
710 XBY(0C000H) = K
720 K = K.OR.128 : REM Enable latch
730 XBY(0C000H) = K
740 K = K.AND.127 : REM Disable latch
750 XBY(0C000H) = K
760 FOR U = 1 TO 100
770 NEXT U
780 RETURN
```

```
Listing 2

5   REM Unmodified circuit with ULN2803
7   XBY(0C000H)= 255 : REM Initial state
    because ULN2803 inverts
10  STRING 100,20 : REM Memory for string
20  K = 12 : GOSUB 700 : REM Init 1
30  FOR Z = 1 TO 4000
40  NEXT Z
50  K = 12 : GOSUB 700 : REM Init 2
60  K = 12 : GOSUB 700 : REM Init 3
70  K = 13 : GOSUR 700 : REM Init 4
80  K = 13 : GOSUB 700 : REM 4-bit interface
90  K = 7 :  GOSUB 700 : REM 2 lines,
    5*7 matrix
100 K = 15 : GOSUB 700 : REM Clear display
110 K = 14 : GOSUB 700
120 K = 15 : GOSUB 700 : REM Cursor home
130 K = 13 : GOSUB 700
14O K = 15 : GOSUB 700 : REM Increment counter,
    do not shift
150 K = 9  : GOSUB 7O0
160 K = 15 : GOSUB 700 : REM display on,
    cursor on, flash
170 K = 0  : GOSUB 700
180 REM———————————————————
250 PRINT.” Max. 32 characters may be entered
    up to arrow  <”
260 INPUT “ Please enter text string to be
    displayed :”,$(1)
270 FOR Z = 1 TO 16
280 D=ASC($(1),Z) : REM Individual char’s,
    supplied by user
290 D = D.XOR.255 : REM ULN 2803 inverts
300 GOSUB 400
310 NEXT Z
320 FOR Y = 16 TO 39 : REM Fill LCD controller
    memory
330 D = 223 : REM not having a display function
340 GOSUB 400
350 NEXT Y : REM with inverted-spaces
360 GOTO 250
370 REM———————————————————
400 REM Subroutine for data output
410 L = D.AND.15 : REM Data of low part
420 H = D.AND.240 : REM Data of high part
430 H = H/16 : REM shift high data
440 H = H.OR.16 : REM Enable writing
450 XBY(0C000H) = H
460 H = H.AND.127 : REM Enable latch
470 XBY(0C000H) = H
480 H = H.OR.128 :REM Disable latch
490 XBY(0C000H) = H
500 FOR U = 1 TO 100
510 NEXT U
520 L = L.OR.16 : REM Enable writing
530 XBY(0C000H) = L
540 L = L.AND.127 : REM Enable latch
550 XBY(0C000H) = L
560 L = L.OR.128 : REM Disable latch
570 XBY(0C000H) = L
580 FOR U = 1 TO 100
590 NEXT U
600 RETURN
610 REM———————————————————
700 REM Subroutine for command outputting
710 K = K.OR.48 : REM Enable command and
    writing
720 XBY(0C000H) = K
730 K = K.AND.127 : REM Enable latch
740 XBY(0C000H) = K
750 K = K.OR.128 : REM Disable latch
760 XBY(0C000H) = K
770 FOR U = 1 TO 100
780 NEXT U
790 RETURN
```

```
Listing 3 (8-bit interface)

10  STRING 100,20   : REM Memory for string
20  XBY(0C000H)=56  : REM 8-bit interface,
    2 lines, 5*7 matrix
30  XBY(0C000H)=15  : REM Display on,
    cursor on, flashing on
40  XBY(0C000H)=1   : REM Clear display
50  PRINT “ Max. 16 characters may be entered
    up to arrow  <”
60  INPUT “ Please enter text string to be
    displayed :”,$(1)
70  FOR Z = 1 TO 16
80  A=ASC($(1),Z) : REM individual char’s,
    supplied by user
90  XBY(0C002H)=A : REM Transmit them!
100 NEXT Z
110 FOR H = 16 TO 39 : REM Fill LCD controller
    memory
120 XBY(0C002H)=32 : not having a display
    function
130 NEXT H : REM with dummy characters
140 GOTO 50
```